Homework # 3:

Note: Please circle your answers as appropriate!

Do ***NOT*** use R for problems 1 - 4.

1) You have 2 rubies (=R) and 9 diamonds (=D)

(a) If you line them up in a row, how many different arrangements can you get?

For example, RRDDDDDDDDD is one arrangement, RDRDDDDDDDD is another arrangement, and so on.

You do NOT want to actually list all possible arrangements - instead, use what you learned in class to calculate this.

(b) Now suppose you have 9 rubies and 2 diamonds. How many different arrangements can you get? Does this make sense?

(c) How many arrangements do you have for 1 ruby and 10 diamonds?

(d) How many arrangements do you have for 0 rubies and 11 diamonds?

2) About 29% of American households own a cat. If Y = number of households with cats in a sample of $n = 7$ households, calculate the following probabilities:

a) $\Pr\{Y = 2\}$ b) $\Pr\{Y = 3\}$ c) $\Pr\{Y < 1\}$ d) $\Pr\{Y > 0\}$
(hint - use (c) to answer this and ***make sure you know why this works****)*

e) You have at least two households with a cat in your sample ($= \Pr\{Y > 1\}$)

3) Mendelian genetics predict that in in peas 75% of offspring plants should have purple flowers if both parents are heterozygous (the rest are white). You mate two adults plants and get 12 pea plants.

Calculate the following:

a) The probability of 8 purple flowers. b) The probability of 4 *white* flowers.

c) The probability that 50% of the flowers will be white.

(Comment: You do not need to know *anything* about genetics to solve this problem. If you're trying to do things like use punnet squares, you're doing things the wrong way.)

4) Refer to problem 3. Suppose you only had 4 plants. Calculate the probability of every possible outcome (you need to do five calculations).

5) Let's learn some miscellaneous R (see instructions at end).

(a) Calculate 73! (b) Calculate 1237!

(c) Refer to problem (1). Suppose you have 37 rubies and 78 diamonds. How many different arrangements can you get?

(d) Refer to problem (1). Suppose now you have 1 ruby and 3,219 diamonds. How many different arrangements can you get?

6) Let's do some more R....

Suppose you have the following situation:

You have a large jar of beans, 32% black, 68% white. You take a sample of 12 beans.

a) Use R to figure out the probability for every single possible outcome. In other words, you need to calculate:

Pr{0 black beans in 12 trials}
Pr{1 black bean in 12 trials}
.
. etc.
.
Pr{12 black beans in 12 trials}

b) Once you have this information, you need to plot a barplot that shows you what this distribution looks like.

As you might guess, you'll need to use the binomial (in R) to solve this problem. Make sure you present all your results (you should have a list of ***probabilities and the barplot***).

Some R instructions to help you are on the next page. Instructions for copying graphs in/from R are also included at the end.

For R problems: be prepared to sketch or list your results. You may also be asked to show your graphs to the class using zoom or blackboard.

Never just hand in a R printout: Clean it up:

Clearly label everything, number your problems, circle your answers, annotate, get rid of stuff that's not needed , etc.

You will not do very well if all you do is hand in a printout!

Due in recitation the week of February 17th.

R commands:

Caution: Be careful copying and pasting commands into R. Some characters may not copy correctly and give you error messages. This is particularly true for quotes. R does not recognize quotes that look like “, only quotes that are straight like this: " . I tried to fix this below, but can't guarantee I caught everything.

Factorials:

Calculating factorials is pretty easy in R. Here's an example calculating 27!:

```
factorial(27)
```

and R should return 1.088887e+28

Binomial coefficients:

Using the binomial coefficient in R is only slightly more complicated since we need two numbers. Suppose we wanted to calculate $\binom{45}{13}$

In R we would do (make sure you put *n* first, or you'll just get 0):

```
choose(45,13)
```

and R should return 73006209045

Binomial probabilities:

Here's an example using a sample of 8 beans from a jar with 32% black beans. You should be able to modify the following as needed to answer question 6:

For probabilities:

You want probabilities for 0 through 8 beans. First we need to create *y* with 8 numbers:

```
y <- c(0:8)
```

This command says to give “y” the numbers 0 - 8 in sequence; technically “0:8” tells R it's a sequence of numbers, and the “c” out front means to combine all the numbers. The “c” isn't really needed (try it!) this time, but it is required in many similar situations so it's a good habit to get into to.

Now we just “feed” our *y* into the binomial function to get our answers:

```
p <- dbinom( y, size = 8, prob = .32)
```

This should give you “p” with all the binomial probabilities (they'll be in order from 0 to 8; type “p” to get/see the probabilities)

If you want to make it look nice, you can do:

```
pr <- data.frame(y,p)
pr
```

to make it look even better, follow this with:

```
print(pr,row.names = FALSE)
```

The data.frame command combines the variables you list (in this case, y and p) into a single data set. Unfortunately if you then type "pr" R will insist on printing row names as well. If you want to get rid of the row names (they're not needed), you need to use the print statement as given above.

Barplots:

To get a barplot, you simply do:

```
barplot(p)
```

This won't look very nice, so you should try to improve it:

```
barplot(p,names.arg = y)
```

Here "names.arg" is the variable that holds the labels you want to put on the x axis (remember we put the numbers 0 - 8 into y). Just try it - you'll see how it works.

If you want to improve your graph even more, you can do (review the caution statement above if you have trouble copying and pasting this):

```
barplot(p,names.arg = y, ylab = "frequency", xlab = "number of dark beans")
```

Or even fancier:

```
barplot(p,names.arg = y, ylab = "frequency", xlab = "number of dark beans", col =
"blue")
```

(Instructions for exporting/saving graphs are on the next page)

To copy/save graphs generated in R or RStudio:

You're going to want to save your graphs or copy them into a word processor so that you can hand them in or refer to them during presentations.

If you are using RStudio:

(If you're given a choice of format to use as you follow these instructions (usually under Windows), choose "metafile")

Make sure the graph you want is visible in the graphics window.

Click on "export" near the top of the plot window.

Select "Copy Plot to Clipboard"

The click on "Copy Plot"

Open your favorite word processor and paste the graph into your document.

Comment: copying graphs this way doesn't always give you the best looking graphs (particularly if you're using Linux). Saving the plot as a high resolution image and then inserting this into your document often gives better results (see instructions for "not using Rstudio" below).

Another possibility is to save the graph as a pdf, and then import this into your document (most recent word processors can handle pdf graphics imports). This usually gives very high quality graphs.

If for some strange reason you're not using RStudio (or want to do this using commands):

Windows:

In R, right click the graph, select "copy as metafile" (don't use bitmap), then open your favorite word processor and paste the graph.

Mac OS:

You should be able to copy the graph (make sure the graph window is active) from the menu, and then simply paste the graph into Word (or whatever word processor you use). If this doesn't work for some reason, the Linux instructions will work (they'll work with Windows as well).

Linux:

This is a bit complicated. A simple way to do it is to use the print screen key, but it'll look horrible. Here's one example of getting good looking graphs:

1) generate the graph on-screen (just as usual) to make sure it looks right.

2) type "`jpeg()`" in the command line. *There are actually several formats you can pick but jpeg is probably the easiest.*

3) generate the graph again (make the graph again). *You'll notice that nothing seems to happen*. That's okay. It's writing the graph to a jpeg file.

4) now type "`dev.off()`" on the command line.

5) the graphics file should now be in your home directory. It'll have a name like "Rplotxxx.jpeg", where xxx is some number. You can always sort by modification date in your file browser to find it quickly.

6) You should now be able to insert or copy the file into your text document (e.g. Word, LibreOffice, or whatever you're using).

7) The jpeg may not look terrific (it'll look a lot better than "print-screen"). You can increase the resolution by doing (in step 2 above):

```
jpeg(width = 1000,height = 1000)
```

The jpeg command defaults 480 x 480, which isn't that great. You can also use the jpeg command to give it a filename that makes sense, if you want:

```
jpeg(filename = "your-file-name",width = xxx, height =
xxx)
```